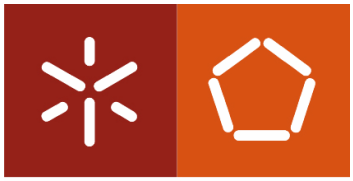

Departamento de Informática
Licenciatura em Engenharia Informática

31 de Maio de 2015

Universidade do Minho
Escola de Engenharia

Conhecimento sub-simbólico

Redes Neuronais Artificiais

Trabalho Prático 3

Sistemas de Representação de Conhecimento e Raciocínio

Grupo 15

Orlando José Gomes Martins Costa a67705

Paulo Ricardo Cunha Correia Araújo a58925

Rui Pedro Azevedo Oliveira a67661

Resumo

O presente documento consiste num relatório de análise à utilização de sistemas sub-simbólicos na representação de conhecimento e no desenvolvimento de mecanismos de raciocínio, nomeadamente Redes Neuronais Artificiais, para a resolução de problemas no domínio da análise de biométricas comportamentais e a sua importância na identificação e classificação de fadiga física e/ou mental. Na fase inicial do relatório são apresentados e analisados os diferentes dados obtidos durante o treino e teste das redes neuronais, assim como as diversas configurações de nodos intermédios, algoritmos de aprendizagem e tamanhos de *input*. É também realizada uma análise à relevância das variáveis sob as quais a rede depende e opera, sendo comparados os resultados obtidos com o conjunto de variáveis inicial e com o conjunto de variáveis presumidas relevantes.

Após configurada a rede, é efetuada sobre esta diferentes testes com o intuito de avaliar os resultados num intervalo de resposta lógico, identificando-se a existência ou ausência de fadiga para um dado *input*.

Finalmente, são realizadas diversas tentativas de *clustering* de forma a encontrar uma escala de fadiga adequada ao problema e aos dados em causa, que diminua a percentagem de erro entre os resultados obtidos e os resultados reais nos testes efetuados à rede neuronal.

O trabalho é desenvolvido sob a plataforma "*R Studio*", que permite facilmente a utilização da linguagem R para análise de *datasets* e a inclusão de diversas bibliotecas que enriquecem a forma como os dados são apresentados e obtidos. Os resultados obtidos são também suportados pela utilização da ferramenta WEKA, que abstrai parte do processo de análise de dados.

O relatório termina com a apresentação dos resultados gerais obtidos e conclusões retiradas, sendo abordados os problemas que emergiram durante a sua realização.

Índice

Introdução

Redes Neuronais Artificias (**RNAs**) constituem uma família de modelos de aprendizagem estatísticos baseados em redes neuronais biológicas (tais como o sistema nervoso humano), e são usados para estimar ou aproximar funções que dependem de um grande número de *inputs*. **RNAs** são geralmente apresentadas como sistemas de neurónios interligados, capaz de trocar "mensagens" entre si. Estas conexões possuem pesos numéricos que podem ser modificados com base na "experiência" da rede, permitindo que as **RNAs** sejam capaz de se adaptar e aprender com os *inputs*.

Neste contexto, são analisados o impacto e peso que um conjunto de biométricas comportamentais possui na deteção de fadiga mental de um dado individuo. Como suporte é utilizado um conjunto de dados globais com experiências anteriores que permitem classificar um comportamento, por recurso a **RNAs** e ao ambiente de análise de dados R.

É estudada a identificação dos 7 níveis de fadiga predefinidos através do treino de uma **RNA** utilizando como input os dados fornecidos, com o intuito de classificar o nível de fadiga através de um conjunto valores correspondentes às variáveis fornecidas (métricas captadas). O objetivo consiste então na estimação do nível de fadiga de um individuo, assim como a existência ou não desta, com base na forma como opera o seu computador. As variáveis das quais depende o treino possuem então um papel importante na determinação no nível de fatiga, e o peso que cada uma destas variáveis tem na resposta final deve ser testado.

Analogamente à importância que as variáveis possuem, o intervalo no qual a fadiga é classificável é também relevante na obtenção de resultados aceitáveis, sendo que este intervalo deve ser estudado com a intenção de produzir *clusters* de dados similares entre sí, e que por isso, são divisíveis e capazes de ser agrupados conforme.

O objetivo final do trabalho é então apresentar uma análise detalhada do impacto que os componentes presentes no treino da rede neuronal possuem no resultado final desta e no erro produzido, assim como a capacidade que a rede tem na previsão de níveis de fadiga apropriados.

Identificação de níveis de fadiga

Um dos objetivos principais a alcançar consiste no treino e teste de uma rede neuronal artificial, tornando-a capaz de estimar o nível de fadiga partindo de *input* que consiste nas diversas variáveis biométricas fornecidas. Assim, é possível em R, através da plataforma "*R Studio*" e da biblioteca "*neuralnet*", criar uma rede neuronal, fornecer-lhe *input* e testar diferentes configurações de forma a comparar os resultados de cada uma. Neste sentido, é feita a análise do impacto que os diferentes componentes e variantes que constituem o treino de uma rede neuronal possuem sobre o resultado final.

Treino da rede

Algoritmos

Existem diversos algoritmos que definem a forma de aprendizagem de uma rede neuronal, isto é, a forma como os seus diferentes nodos interagem e como variam os pesos destes. Estes algoritmos possuem um impacto no resultado final e consistem nas seguintes variantes:

- *rprop+* e *rprop-*: Algoritmos de *backpropagation* (pesos dos nós alterados com base no resultado produzido) resiliente, com e sem retrocesso dos pesos, respetivamente. Consistem numa heurística para aprendizagem supervisionada em redes neuronais artificiais *feedforward*.

- *sag* e *slr*: Algoritmos que induzem o algoritmo globalmente convergente (*grprop*). Estes são baseados nos algoritmos de *backpropagation* resiliente, com a variante que modificam a componente da taxa de aprendizagem (associada com o gradiente absoluto mais baixo ou com a taxa de aprendizagem mais baixa, respetivamente).

Nodos intermédios

Diferentes estruturas de redes neuronais alcançam diferentes resultados, tanto em erro, como em tempo de execução. Assim, uma rede neuronal com mais neurónios do que outra pode chegar ao resultado esperado em menos tempo, mas esse resultado pode não estar de acordo com o esperado. Com o objetivo de testar as diferentes constituições de redes neuronais e o seu impacto sobre o output, são testadas diversas configurações dos nodos que as constituem:

- (3): 3 nodos intermédios numa só camada.
- (10): 10 nodos intermédios numa só camada.
- (20, 10): 30 nodos intermédios divididos por duas camadas.
- (40, 20): 60 nodos intermédios divididos por duas camadas.

Uma representação gráfica de cada topologia pode ser encontrada em anexo (anexo A).

O resultado que se espera obter com a rede neuronal de (40,20) nodos intermédios pensa-se ser aceitável em termos do erro produzido durante a sua aprendizagem (tendo em conta as diferentes variáveis de input), pelo que aumentar ainda mais a complexidade da rede neuronal não seria vantajoso. Da mesma forma, a rede neuronal de apenas 3 nodos intermédios crê-se ser insuficiente para alcançar um erro significativo durante a aprendizagem da rede neuronal artificial, no entanto pode produzir resultados apropriados para o contexto. Certos estudos sugerem que a utilização de mais de 3 nodos intermédios não é vantajosa, facto este analisado e testado neste relatório. As restantes configurações servem como modelo comparativo das configurações referidas anteriormente.

Diferentes tamanhos e ordenações de *input* de treino

O tamanho do *dataset* utilizado para treinar a rede influência os resultados que esta alcançará no momento de a testar. Teoricamente, quanto maior e mais diversificado for o *dataset*, mais consistente será a rede neuronal, e melhor responderá aos testes efetuados. Assim, dado que o *dataset* fornecido possui apenas cerca de 800 registos, este é dividido entre *datasets* de treino e teste de rede, com diferentes tamanhos. A ordem pela qual estes testes participam no treino da rede neuronal afeta também o resultado final e os cálculos efetuados por esta, pelo que são testadas diferentes ordenações de *datasets* de treino (*datasets* original e invertido).

Procede-se então à análise dos resultados obtidos com as diversas configurações que é possível obter com a variação dos componentes descritos.

Testes realizados

O treino das redes neuronais e testes é efetuado recorrendo à linguagem R, suportada pela plataforma "*R Studio*" e pela biblioteca "***neuralnet***". Como métricas para análise de resultados, são analisados o erro produzido no treino das redes, assim como o "*Root Mean Square Error (rmse)*" e "*Percent BIAS (PBIAS)*", sendo estes últimos suportados pela biblioteca "***hydroGOF***", e que permitem analisar de forma analítica a diferença média entre os resultados obtidos nos testes efetuados e os resultados esperados, e a tendência média que os resultados possuem comparativamente ao esperado (valores mais elevados ou mais baixos).

Análise de resultados

Através de uma análise breve à tabela de resultados presente em anexo (Anexo B), é possível concluir diversos factos e assumir algumas teorias:

- Existem diversas ocasiões nas quais o treino da rede não produziu qualquer resultado quantificável (não convergiu). Isto pode ocorrer devido a diversos fatores, tais como o peso inicial atribuído aos nodos, o número máximo de iterações não ser suficiente ou o algoritmo requer mais capacidade de processamento do que os outros. Como se pode observar, a não convergência é mais comum nos algoritmos *sag* e *slr* (globalmente convergentes), provavelmente devido a estes requererem maior carga computacional. De notar também que a configuração de (20,10) nodos intermédios em caso nenhum converge, pelo que não será uma considerada como uma configuração válida.

- Em média as *RNAs* cujas configurações consistem em 3 nodos intermédios demoram muito menos tempo do que as configurações com duas camadas (40,20) de nodos intermédios, sendo necessários menos passos para convergir. Estas configurações mais simples possuem também em média um RMSE ligeiramente menor sendo que os valores médios de PBIAS não são conclusivos. No entanto, as configurações mais complexas obtêm um erro de aprendizagem em média muito menor às configurações mais simples.

- Os resultados em PBIAS são mais elevados no *dataset* invertido do que no original, assim como o erro de aprendizagem das redes e o tempo até convergirem. Isto talvez possa ser explicado devido à distribuição pouco uniforme dos diferentes tipos de *Tasks* no ficheiro de input, dado que o ficheiro original possui uma maior concentração de *Tasks* do tipo 1 na 1ª metade da lista de registos. Assim, a rede será 1ºprimeiro treinada com estes registos possuindo uma tendência para estimar registos do tipo 1, quando na realidade o ficheiro de teste possui uma maior diversidade de registos do tipo 2 e 3. O caso inverso ocorre no ficheiro invertido, causando diferentes valores no cálculo dos pesos nas sucessivas iterações.

- Os resultados obtidos pelo algoritmo *rprop-* são ligeiramente melhores que os produzidos pelo algoritmo *rprop+*, no entanto esta diferença não é significativa.

- As *RNAs* que utilizam os algoritmos baseados em *grprop* raramente convergem quando possuem mais do que 3 nodos intermédios. Estes, nas ocasiões em que convergem, não produzem resultados significativamente diferentes do que os obtidos com os algoritmos baseados em *backpropagation*.

Estrutura RNA	Média Erro	Média RMSE	Média PBIAS
3	227.2482252	1.763169147	-12.64782609
10	131.0173075	1.524007605	-8.316666667
20, 10	NA	NA	NA
40, 20	2.434903571	2.143438802	-6.957142857

Tabela: Média Global dos Resultados das diferentes redes.

Desta forma são selecionadas as melhores configurações de treino de *RNAs* produzidas por cada algoritmo, que servirão de base para comparações e otimizações futuras:

N	Tamanho Input	Algoritmo	Nodos Intermédios
1	400(invertido)	rprop+	40,20
2	500	rprop-	40,20
3	600	sag	3
4	400	slr	3

Tabela: Melhores configurações de RNA's.

Análise de variáveis relevantes

Apesar de que os registos fornecidos apresentam 9 variáveis que caracterizam o nível de fadiga, é possível que nem todas as variáveis possuam o mesmo peso na decisão do output final, e então, podem ser prescindíveis, não apenas simplificando os cálculos da rede em causa, mas melhorando o treino e o output gerado pelos testes. A medição de biométricas comportamentais possui diversas variantes cuja relevância deve ser estudada. Neste sentido, é então estudado o peso que as variáveis possuem, e são comparados os resultados obtidos anteriormente com os registos originais relativamente aos resultados obtidos com os registos mais relevantes.

Ferramenta WEKA

Inicialmente foi realizada uma abordagem que consistia na utilização de bibliotecas R, que permitem a análise do peso que as diferentes variáveis possuem sobre a estimativa do valor da variável de output numa rede neuronal artificial. No entanto, esta abordagem é limitada e produz resultados inconsistentes. Neste sentido, é dado uso a uma nova ferramenta de estudo de dados denominada "***Weka***", que possui um conjunto de algoritmos de *machine learning* que permitem a execução de tarefas de *data mining*. Mais especificamente, são exploradas as suas capacidades de processamento de dados, que nos permitem classificar as variáveis de maior peso na determinação de um dado output, evitando assim a maçadora tarefa de 'tentativa e erro' (testar as diversas combinações de variáveis que produzem o melhor resultado).

Análise de resultados

Com o intuito de avaliar então o mérito que cada variável possui na determinação da fadiga, seleciona-se o atributo *FatigueLevel* como variável de output a analisar, e obtêm-se os seguintes resultados:

Search Method: Best first.

Start set: no attributes

Search direction: forward

Stale search after 5 node expansions

Total number of subsets evaluated: 45

Merit of best subset found: 0.233

Attribute Subset Evaluator (supervised, Class (numeric): 9 FatigueLevel): CFS Subset Evaluator

Including locally predictive attributes

Selected attributes: 1,2,5,10 : 4

Performance.KDTMean

Performance.MAMean

Performance.DDCMean

Performance.Task

Assim, os resultados indicam que o subconjunto de variáveis que possui maior peso na determinação da variável *FatigueLevel* consiste nos atributos *Performance.KDTMean, Performance.MAMean, Performance.DDCMean, Performance.Task*.

De forma a avaliar o impacto que o novo conjunto de variáveis possui relativamente ao conjunto original, são testadas as redes cuja configuração obteve melhor resultado nos testes realizados anteriormente, sendo a sua função de aprendizagem constituída agora pelas variáveis do novo conjunto, e comparam-se os valores obtidos. O mesmo teste é efetuado com as variáveis restantes, tirando-se seguintes conclusões:

- Apenas o treino correspondente à *RNA* com o algoritmo *slr* convergiu. Isto pode ser devido a diversos fatores, tal como a possibilidade de que a existência de menos variáveis de input obrigue a *RNA* a efetuar um maior número de passos de forma a convergir e produzir um resultado aceitável.
- Os resultados obtidos na configuração que obteve sucesso com as variáveis relevantes são ligeiramente piores que os resultados originais, no entanto a diferença não é significativa, pelo que o único que se pode concluir é que com apenas 4 das 9 variáveis originais é possível obter os mesmos resultados, confirmando o peso que estas variáveis possuem.
- Como esperado, as *RNAs* relativas às variáveis consideradas menos relevantes produzem resultados bastante inferiores aos obtidos com todas as variáveis, nos casos em que convergem.

Os valores obtidos podem ser consultados na tabela em anexo (Anexo C).

Como é possível observar, os resultados produzidos por cada *RNA* dependem totalmente das suas diversas configurações. Não existe uma configuração ideal para cada *dataset*/situação, pelo que estas devem ser sujeitas a testes de forma a encontrar as que melhores resultados obtêm.

Identificação de existência ou ausência de fadiga

Com o intuito de identificar a existência ou ausência de fadiga dado um determinado *input*, são feitas algumas considerações acerca dos níveis de fadiga mental apresentados. Através da análise da descrição de cada nível, é possível extrair 2 grupos de estados mentais. O 1ºgrupo refere-se a estados onde um dado individuo tem ainda a capacidade de funcionar e raciocinar dentro da normalidade, e é constituído pelos estados de 1 a 3 (inclusive). Quando um dado output estiver entre estes valores, considera-se a ausência de fadiga. De forma análoga, o 2ºgrupo refere-se a estados nos quais o individuo perdeu certa capacidade de concentração e apresenta sinais de cansaço, e é constituído pelos estados de 4 a 7. Quando um dado output estiver entre estes valores, considera-se a existência de fadiga. Para facilitar a compreensão, os valores dos *datasets* foram normalizados para apresentar respostas lógicas, e então, testes cujo resultado pertença ao 1ºgrupo apresentarão um output de 0, e testes cujo resultado pertença ao 2ºgrupo apresentarão um output de 1. Os testes consistem num conjunto de registos obtidos do ficheiro fornecido e de registos criados também especificamente para este propósito, e são aplicados nas 3 redes cujas configurações obtiveram os melhores resultados anteriormente (com o conjunto original de variáveis e com o novo conjunto de variáveis relevantes). Os testes são realizados recorrendo às redes neuronais artificiais disponíveis através da utilização da biblioteca "**neuralnet"** na plataforma "**R Studio"**.

Testes realizados

O out*put* correspondente aos testes realizados consiste em:

- 3 testes com valores retirados do *dataset* original, cujos níveis de fatiga correspondem a 1, 3 e 6, e então devem ter como output 0, 0 e 1, respetivamente.
- 1 teste "extremo superior", com os valores das variáveis todos a 1, e então o seu valor de output deverá ser 1.
- 1 teste "extremo inferior", com os valores das variáveis todos a -1, e então o seu valor de output deverá ser 0.

Obviamente que no caso dos testes extremos, os valores de output podem não ser os esperados, especialmente tendo em conta a inexistência deste tipo de registos no *dataset* de treino.

Análise de resultados

Após efetuados os testes, podem ser derivadas as seguintes conclusões:

- Os testes nos quais é pretendido estimar a ausência de fadiga possuem uma taxa de acerto bastante elevada comparativamente aos testes onde se espera presença de fadiga. Isto pode ser devido ao facto de que existem muitos mais registos que denotam a ausência de fadiga do que a sua presença, pelo que a rede será tendenciosa a avaliar os casos como "ausência de fadiga" (particularmente tendo em conta o facto de que as redes estão treinadas com apenas 400 registos).
- Os testes extremos de fadiga possuem resultados completamente errados, como estava previsto. A ausência de registos no *dataset* de treino que espelhem estes testes torna o seu resultado "aleatório", pelo que de forma a melhorar esta situação seria necessário treinar a *RNA* com registos semelhantes.

Os resultados numéricos podem ser encontrados em anexo (Anexo D).

Otimização da escala de identificação de fadiga

Um dos desafios propostos consiste em encontrar a melhor escala de identificação de fadiga possível, e então, uma escala tal que não apenas diminua o erro dado pelas métricas usadas (p.e. *rmse*), mas que proporcione resultados mais consistentes quando é efetuado o treino e teste da rede neuronal.

Para este efeito é explorada uma técnica muito comum no domínio de *Machine Learning* denominada *clustering*. *Clustering* consiste no agrupamento de registos de dados ou objetos de tal forma que objetos no mesmo grupo (ou cluster) sejam mais similares entre sí do que com elementos de outros clusters.

Número ótimo de *clusters*

Uma breve análise ao *dataset* original permite retirar alguns factos relevantes sobre os dados fornecidos:

FatigueLevel	**Número de registos por nível de fadiga:**
Min. :1.000000	1 - 184
1st Qu.:2.000000	2 - 257
Median :2.000000	3 - 247
Mean :2.484597	4 - 126
3rd Qu.:3.000000	5 - 26
Max. :6.000000	6 - 4

- Apesar do nível de fatiga estar atualmente dividido em 7 níveis, o maior nível observado é o nível 6.
- A média e a mediana são relativamente baixas, o que indica um elevado número de registos nos quais é possível afirmar que não existe fadiga.
- Mais de 50% dos registos encontram-se nos níveis 2 e 3.

Tendo em conta estas afirmações assim como os erros obtidos durante as fase de treino e teste da rede neuronal, é possível pressupor que o número de *clusters* poderá ser otimizado e assim os registos distribuídos pela nova escala de fadiga.

Cálculo do número de *clusters*

Uma abordagem a seguir no cálculo do número de *clusters* a utilizar seria remover o nível 7 do *dataset* original e distribuir os registos por 3 novos clusters. Estes *clusters* corresponderiam então a pares de níveis de fadiga, e permitiriam aumentar a concentração e existência de registos em cada um, assim como reduzir enormemente o erro obtido nos testes. No entanto, esta abordagem baseia-se em suposições, e para evitar a utilização do método de "tentativa e erro", são utilizadas primitivas que permitem obter o resultado esperado analiticamente.

K-means clustering

K-mean é um método de *clustering* comum que consiste na associação de registos com registos similares. Inicialmente escolhem-se k registos aleatórios que servem de centróides iniciais. Assim, a cada iteração associa-se um destes registos ao centróide mais próximo (similar), e recalculam-se os centróides (o centróide consiste na média dos valores associados). No final todos os registos estão contidos num dos k-clusters. Uma das vantagens deste método é que um registo pode não estar permanentemente associado a um dado *cluster*, podendo mudar de cluster se isso melhorar a solução.

Em R existe a possibilidade de usar a função ***kmeans*** que permite aplicar o algoritmo referido a um dado *dataset*. No entanto, esta função deve também receber como parâmetro o número de *clusters* a formar, pelo que de forma a evitar um número elevado de testes de "tentativa e erro" torna-se necessário averiguar a quantidade de clusters que proporcionará uma melhor solução.

Após uma aprofundada pesquisa, foi encontrada uma biblioteca que possui vários métodos de clustering e cluster *validation*, denominada "**fpc**".Em particular, esta biblioteca possui uma função denominada "**pamk**", capaz de estimar o número de *clusters* ótimo baseando-se em primitivas de *k-means clustering* (basicamente efetua a tentativa e erro).

Assim, dado um intervalo aceitável de número de *clusters*, é utilizada uma função denominada ***wssplot*** que através de sucessivas utilizações da função ***kmeans***, gera um gráfico representativo do número apropriado de clusters relativos ao *dataset*:

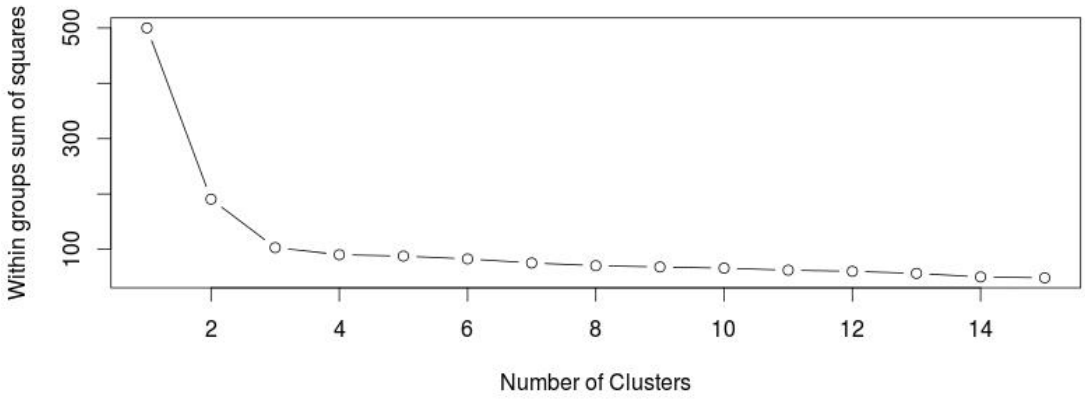

Figura: Número de clusters que produz melhores resultados.

O decréscimo na curva da função indica a tendência que estes dados possuem em formarem 3 grupos de registos com semelhanças. Assim, o output reforça a teoria de que o número de clusters mais adequando é formado por 3 clusters. Através da função pamk podemos reforçar esta teoria, dado que o output indica também a preferência por 3 clusters.

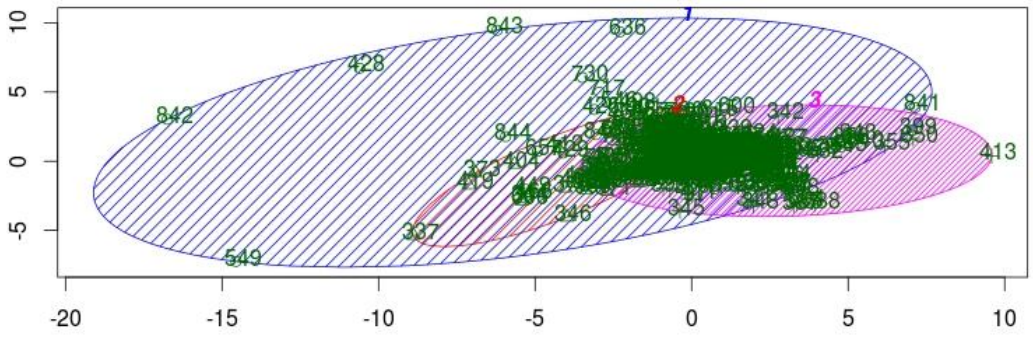

Figura: Clusters identificados.

Finalmente através da função ***kmeans*** podemos também atribuir aos registos o atributo numérico do *cluster* no qual estes se inserem e obter um novo *dataset* com o qual podemos reavaliar os testes realizados anteriormente.

Não obstante os resultados obtidos, estes foram validados com a utilização da ferramenta WEKA, cuja solução final consiste também na utilização de 3 *clusters* (Anexo E).

Clustered Instances	
0	116 (14%)
1	84 (10%)
2	35 (4%)
3	458 (54%)
4	151 (18%)

Análise de resultados

Os resultados obtidos anteriormente são agora revalidados de acordo com o novo conjunto de *clustering*, sendo considerada a existência de fadiga apenas no nível 3 (dado o baixo volume de registos originais com um nível de fadiga elevado). Analisando os resultados obtidos (Anexo F), podemos derivar as seguintes conclusões:

- Para todas as configurações, os resultados melhoraram imensamente, obtendo inclusive estimativas completamente corretas (rmse a 0). Estes testes foram repetidos de forma a assegurar a sua validade, e confirma a importância da divisão correta em grupos de um dado conjunto de registos.

- A estimação da presença ou ausência de fadiga produziu valores pouco corretos. Isto pode ser explicado por uma má classificação dos níveis nos quais se denota fadiga ou não.

Conclusão

Durante a realização do trabalho foi analisado o impacto que os diversos componentes presentes no treino de uma rede neuronal artificial possuem no resultado produzido por esta e nos teste efetuados. Estes componentes não apenas afetam os resultados produzidos, como também influenciam o tempo que demora a treinar uma dada *RNA*, e o erro resultante da sua aprendizagem. Foi também possível denotar que existem certas variáveis que possuem maior peso na determinação do resultado final, sendo que estas podem inclusive tornar as outras desnecessárias na estimação de resultados, propiciando um treino de *RNAs* mais simples e efetivo. A determinação de um número de *clusters* apropriado é também fulcral na obtenção de resultados significativos, sendo que foi possível obter uma melhoria relativamente elevada nos resultados obtidos durante o progresso deste trabalho. Não obstante, as ferramentas utilizadas facilitaram imenso a análise dos diversos outputs obtidos, e constituem também uma componente chave para obter sucesso na determinação das melhores formas de abordar um dado conjunto de dados.

Em forma de crítica final, apesar dos problemas encontrados, o grupo sente-se confiante na sua habilidade de cumprir os desafios propostos e crê ter atingido o objetivo subjacente na realização do trabalho, tendo realizado uma análise completa das funcionalidades das redes neuronais através da utilização de ferramentas diversas e de dados relacionados com biométricas comportamentais.

Anexo A - Topologia das RNAs

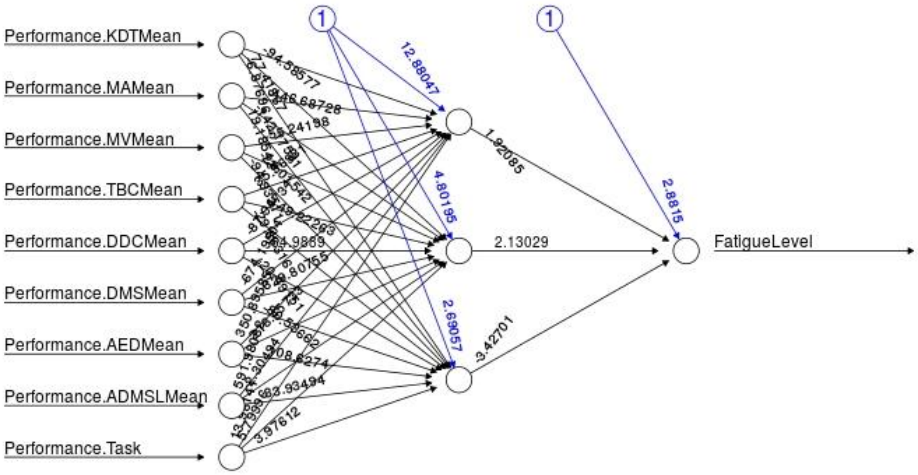

Figura: RNA com 3 nodos intermédios.

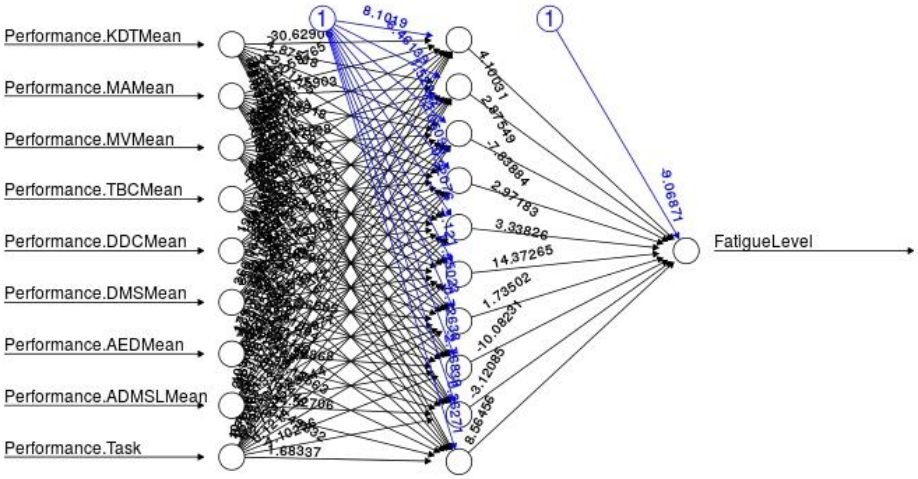

Figura: RNA com 10 nodos intermédios.

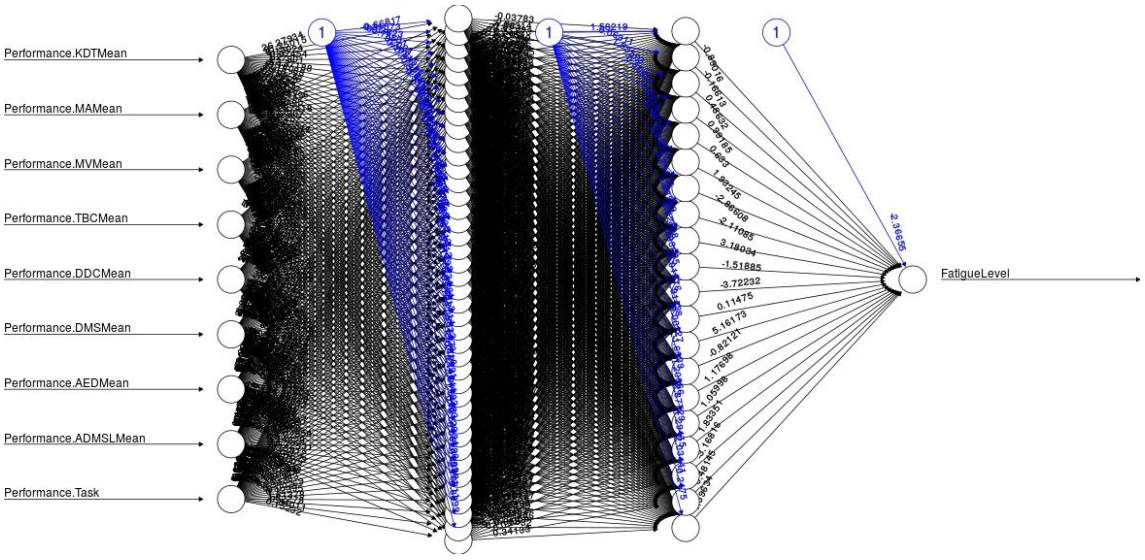

Figura: RNA com (40,20) nodos intermédios.

Anexo B – Resultados de identificação de níveis de fadiga

Data file	Input Treino	Input Teste	Algoritmo	N. Int.	Passos	Erro	Tempo	RMSE	PBIAS
Original	400	400+	*rprop+*	3	42757	130.04491	17.96 secs	1.271659198	-4.8
				10	42481	77.67486	26.28 secs	1.465058496	5.5
				20, 10	max	nao convergiu			
				40, 20	9477	0.00327	39.97 secs	1.872032186	14.7
			rprop-	3	39498	160.42751	17.76 secs	1.282241843	-8.3
				10	47263	75.96196	25.65 secs	1.444155665	7.3
				20, 10	max	nao convergiu			
				40, 20	7883	0.0083	29.12 secs	2.476011032	19.6
			sag	3	max	nao convergiu			
				10	72799	80.13497	40.09 secs	1.50599403	12.2
				20, 10	max	nao convergiu			
				40, 20	10888	0.01479	44.6 secs	2.388504753	11.8
			slr	3	7729	137.91119	3.21 secs	1.249324142	-4.5
				10	max	nao convergiu			
				20, 10	max	nao convergiu			
				40, 20	max	nao convergiu			
	500	300+	*rprop+*	3	25409	192.24517	12.99 secs	1.30183234	-11.6
				10	39734	111.25297	26.76 secs	1.427512066	-2.3
				20, 10	max	nao convergiu			
				40, 20	11513	0.00256	1.01 mins	1.896293815	-16
			rprop-	3	31104	182.47358	20.59 secs	1.310733862	-12.4
				10	30197	130.38868	19.88 secs	1.965546258	-8.2
				20, 10	max	nao convergiu			
				40, 20	12975	0.00755	55.89 secs	1.847374	-7.1

			sag	3	33853	176.89571	16.3 secs	1.330544564	-11.9
				10	max	nao convergiu			
				20, 10	max	nao convergiu			
				40, 20	max	nao convergiu			
			slr	3	56685	185.20023	27.03 secs	1.381985427	-16
				10	max	nao convergiu			
				20, 10	max	nao convergiu			
				40, 20	max	nao convergiu			
	600	200+	*rprop+*	3	27026	258.5041	14.15 secs	1.281968261	3.2
				10	max	nao convergiu			
				20, 10	max	nao convergiu			
				40, 20	40789	0.02461	4.46 mins	2.104834404	-7
			rprop-	3	25409	259.6957	12.74 secs	1.27234133	1.7
				10	36644	193.11213	28.93 secs	1.293109101	2.3
				20, 10	max	nao convergiu			
				40, 20	24228	0.02095	2.24 mins	1.938606902	-2.5
			sag	3	69037	264.67692	36.76 secs	1.22808661	-0.8
				10	max	nao convergiu			
				20, 10	max	nao convergiu			
				40, 20	45548	0.01496	12.12 mins	2.468655969	-24.1
			slr	3	99345	254.74182	1.41 mins	1.234742955	5.7
				10	max	nao convergiu			
				20, 10	max	nao convergiu			
				40, 20	max	nao convergiu			
Invertido	400	400+	*rprop+*	3	9163	223.50889	4.14 secs	1.246617044	-4.4
				10	65274	110.07749	39.72 secs	1.423736994	-10.6
				20, 10	max	nao convergiu			

				40, 20	21366	5.4515	1.54 mins	1.861777703	-2.8
			rprop-	3	16076	212.18297	6.85 secs	3.199098972	-39.1
				10	44093	99.00584	24.42 secs	1.438686839	-9.1
				20, 10	max	nao convergiu			
				40, 20	23403	5.97252	1.53 mins	2.419422163	1.3
			sag	3	56273	200.67215	26.27 secs	1.312620651	-8.1
				10	max	nao convergiu			
				20, 10	max	nao convergiu			
				40, 20	29607	5.52216	2.01 mins	2.537999496	-5.3
			slr	3	44652	204.40713	17.94 secs	10.9819672	-71.7
				10	max	nao convergiu			
				20, 10	max	nao convergiu			
				40, 20	max	nao convergiu			
	500	300+	*rprop+*	3	17551	249.63144	8.37 secs	1.212819126	-14.8
				10	47101	167.2267	49.92	1.562645342	-26.5
				20, 10	max	nao convergiu			
				40, 20	41435	5.46008	3.41 mins	1.86148228	-28.4
			rprop-	3	14861	250.13816	6.64 secs	1.09968283	-3.8
				10	55778	156.36509	36.14 secs	1.773513134	-21.7
				20, 10	max	nao convergiu			
				40, 20	48324	5.57999	3.56 mins	1.910040789	-20.8
			sag	3	66994	248.05374	35.94 secs	1.383036767	-19.4
				10	max	nao convergiu			
				20, 10	max	nao convergiu			
				40, 20	max	nao convergiu			
			slr	3	15417	276.59517	7.33 secs	1.153861061	-14.2
				10	max	nao convergiu			

				20, 10	max	nao convergiu			
				40, 20	max	nao convergiu			
	600	200+	*rprop+*	3	30317	282.02222	28.33 secs	1.20449976	-16.6
				10	77061	179.75695	1.14 mins	1.531098395	-24.8
				20, 10	max	nao convergiu			
				40, 20	89641	6.00541	10.47 mins	2.425107738	-30.8
			rprop-	3	21021	289.89534	20.56 secs	1.161189328	-12.8
				10	37031	191.25005	29.72 secs	1.457034941	-23.9
				20, 10	max	nao convergiu			
				40, 20	max	nao convergiu			
			sag	3	15326	286.12615	20.17 secs	1.14161232	-7.9
				10	max	nao convergiu			
				20, 10	max	nao convergiu			
				40, 20	max	nao convergiu			
			slr	3	42890	300.65898	26.83 secs	1.310424784	-18.4
				10	max	nao convergiu			
				20, 10	max	nao convergiu			
				40, 20	max	nao convergiu			

Tabela: Resultados do treino e teste de RNAs.

Anexo C – Resultados de identificação de fadiga com variáveis mais e menos relevantes

Data file	Input Treino	Algoritmo	N. Int.	Passos	Erro	Tempo	RMSE	PBIAS
Invertido	400	rprop+	40,20	max	não convergiu			
Original	500	rprop-	40,20	max	não convergiu			
	600	sag	3	max	não convergiu			
	400	slr	3	20991	153.71306	12.23 secs	1.278724026	7.5

Tabela: Resultados de treino com as variáveis mais relevantes.

Data file	Input Treino	Algoritmo	N. Int.	Passos	Erro	Tempo	RMSE	PBIAS
Invertido	400	rprop+	40,20	46812	6.01811	6.73 mins	2.342325017	-18.1
Original	500	rprop-	40,20	97245	0.11566	16.7 mins	2.21122927	-12.7
	600	sag	3	max	não convergiu			
	400	slr	3	max	não convergiu			

Tabela: Resultados de treino com as variáveis menos relevantes.

Anexo D – Identificação de existência ou ausência de fadiga

Teste	Data file	Input Treino	Algoritmo	N. Int.	R. Esperado	R. Obtido
Normal -> 1	Invertido	400	rprop+	40,20	0	1
	Original		slr	3	0	0
Normal -> 3	Invertido		rprop+	40,20	0	0
	Original		slr	3	0	0
Normal -> 6	Invertido		rprop+	40,20	1	1
	Original		slr	3	1	0
Teste extremo superior	Invertido		rprop+	40,20	1	0
	Original		slr	3	1	0
Teste extremo inferior	Invertido		rprop+	40,20	0	2
	Original		slr	3	0	0

Tabela: Resultados obtidos nos testes de identificação de existência ou ausência de fadiga.

Anexo E – Output de clustering com a ferramenta WEKA

=== Run information ===

Scheme: weka.clusterers.EM -I 100 -N -1 -X 10 -max -1 -ll-cv 1.0E-6 -ll-iter 1.0E-6 -M 1.0E-6 -K 10 -num-slots 1 -S 100

Relation: exercicio3n

Instances: 844

Attributes: 10

Performance.KDTMean

Performance.MAMean

Performance.MVMean

Performance.TBCMean

Performance.DDCMean

Performance.DMSMean

Performance.AEDMean

Performance.ADMSLMean

Performance.Task

Ignored:

FatigueLevel

Test mode: evaluate on training data

=== Clustering model (full training set) ===

EM

==

Number of clusters selected by cross validation: 5

Number of iterations performed: 2

	Cluster				
Attribute	0	1	2	3	4
	(0.16)	(0.07)	(0.04)	(0.59)	(0.14)
Performance.KDTMean					
mean	0.1106	-0.0072	0.045	0.0066	0.0495
std. dev.	0.258	0.0349	0.1185	0.0223	0.1025
Performance.MAMean					
mean	-0.0099	-0.0209	0.1098	-0.0167	-0.0244
std. dev.	0.1497	0.2154	0.2968	0.0692	0.0833
Performance.MVMean					
mean	-0.0016	-0.0291	0.0935	-0.0059	-0.0055
std. dev.	0.1533	0.1432	0.2637	0.0364	0.0408
Performance.TBCMean					
mean	0.0301	0.0193	0.1542	0.0073	0.0026
std. dev.	0.0951	0.0588	0.3041	0.0642	0.011
Performance.DDCMean					
mean	0.1089	0.1238	0.0398	0.0172	0.241
std. dev.	0.1848	0.1805	0.1538	0.0545	0.2507
Performance.DMSMean					
mean	-0.0026	0.019	0.2229	0	0.019
std. dev.	0.1124	0.0677	0.2887	0.0185	0.0366
Performance.AEDMean					
mean	0.0044	0.0177	0.1692	0.0052	-0.0022
std. dev.	0.0352	0.0772	0.3025	0.0425	0.0152

Performance.ADMSLMean

mean	0.007	0.0452	0.312	-0.0083	0.1103
std. dev.	0.191	0.2002	0.4217	0.1012	0.2063

Performance.Task

mean	1.1314	2.4668	1.8943	1.7141	1.9945
std. dev.	0.3985	0.6221	0.7272	0.7025	0.0817

Time taken to build model (full training data) : 15.89 seconds

=== Model and evaluation on training set ===

Clustered Instances

0 116 (14%)

1 84 (10%)

2 35 (4%)

3 458 (54%)

4 151 (18%)

Log likelihood: 9.4455

Anexo F – Resultados pós-clustering

Data file	Input Treino	Algoritmo	N. Int.	Passos	Erro	Tempo	RMSE	PBIAS
Invertido	400	rprop+	40,20	229	0.005	2 secs	0	0
	500	rprop-	40,20	210	0.01208	1.93 secs	0.07624928517	-0.4
	600	sag	3	max	não convergiu			
Original	400	slr	3	8943	0.00178	5.74 secs	0.06711560552	0.3

Tabela: Resultados de testes efetuados com variáveis originais.

Data file	Input Treino	Algoritmo	N. Int.	Passos	Erro	Tempo	RMSE	PBIAS
Invertido	400	rprop+	40,20	223	0.00444	1.73 secs	0	0
	500	rprop-	40,20	160	0.00263	1.56 secs	0	0
	600	sag	3	max	não convergiu			
Original	400	slr	3	max	não convergiu			

Tabela: Resultados de testes efetuados com variáveis mais relevantes.

Teste	Data file	Input Treino	Algoritmo	N. Int.	R. Esperado	R. Obtido
Normal -> 1	Invertido	400	rprop+	40,20	0	1
	Original		slr	3	0	1
Normal -> 3	Invertido		rprop+	40,20	0	1
	Original		slr	3	0	1
Normal -> 6	Invertido		rprop+	40,20	1	0
	Original		slr	3	1	0
Teste extremo superior	Invertido		rprop+	40,20	1	1
	Original		slr	3	1	0
Teste extremo inferior	Invertido		rprop+	40,20	0	0
	Original		slr	3	0	0

Tabela: Resultados de determinação de existência/ausência de fadiga

com variáveis mais relevantes.